US 20050177580A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2005/0177580 A1

Hilbert et al. (43) Pub. Date: Aug. 11, 2005

(54) SYSTEM AND METHOD FOR CUSTOMIZED DOCUMENT SELECTION

(76) Inventors: **David M. Hilbert**, Palo Alto, CA (US); **Jonathan J. Trevor**, Santa Clara, CA (US)

Correspondence Address:
FLIESLER MEYER, LLP
FOUR EMBARCADERO CENTER
SUITE 400
SAN FRANCISCO, CA 94111 (US)

(21) Appl. No.: **10/777,289**

(22) Filed: **Feb. 11, 2004**

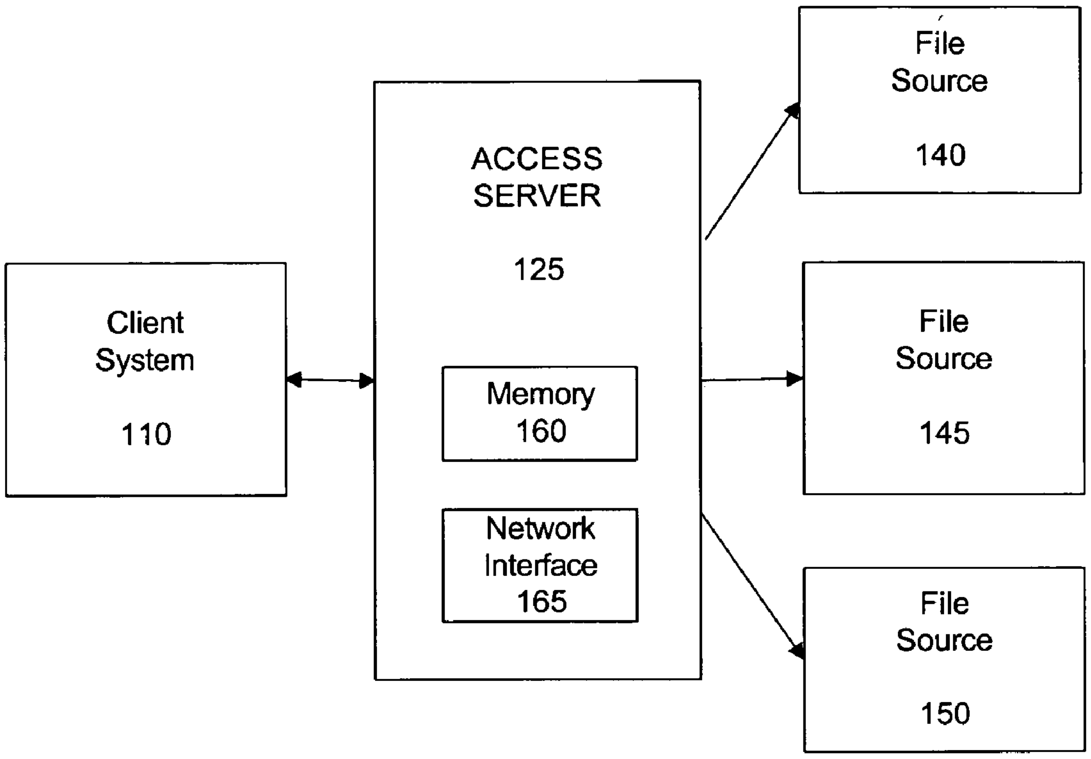
Publication Classification

(51) **Int. Cl.**^{7} **G06F 7/00**; G06F 17/00

(52) **U.S. Cl.** **707/100**

(57) **ABSTRACT**

An access server allows secure access to commonly used files stored on multiple file sources from a variety of client devices. The access server stores user identification and password information for each of a variety of file sources. Upon receiving a login request from a user, the access server extracts a list of file sources associated with the user, and accesses each of those sources. The server is then configured to extract a list of most recently used files by the current user as well as information associated with those files. The server presents to the user an interface including links to his most recently used files.



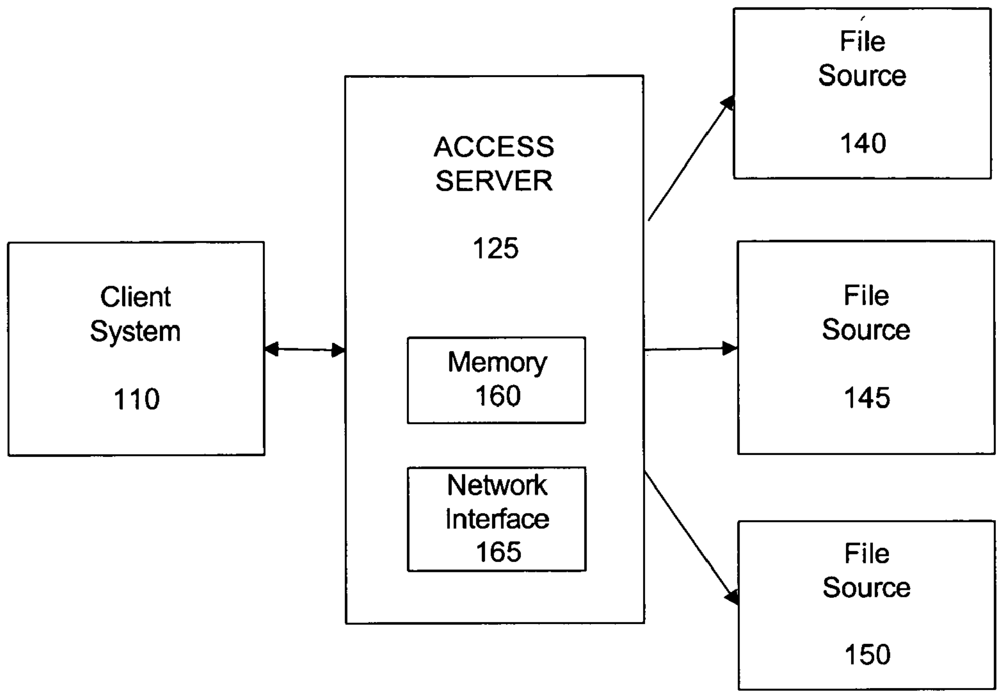

Client System 110
ACCESS SERVER 125
Memory 160
Network Interface 165
File Source 140
File Source 145
File Source 150

Fig. 1

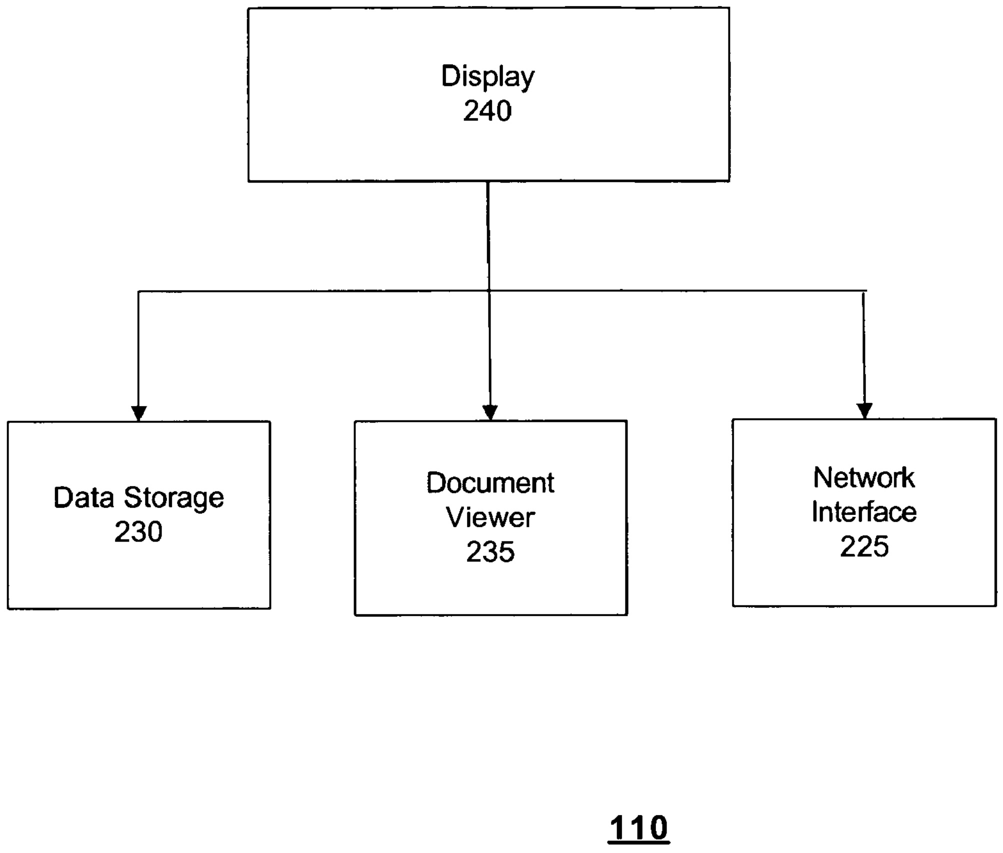
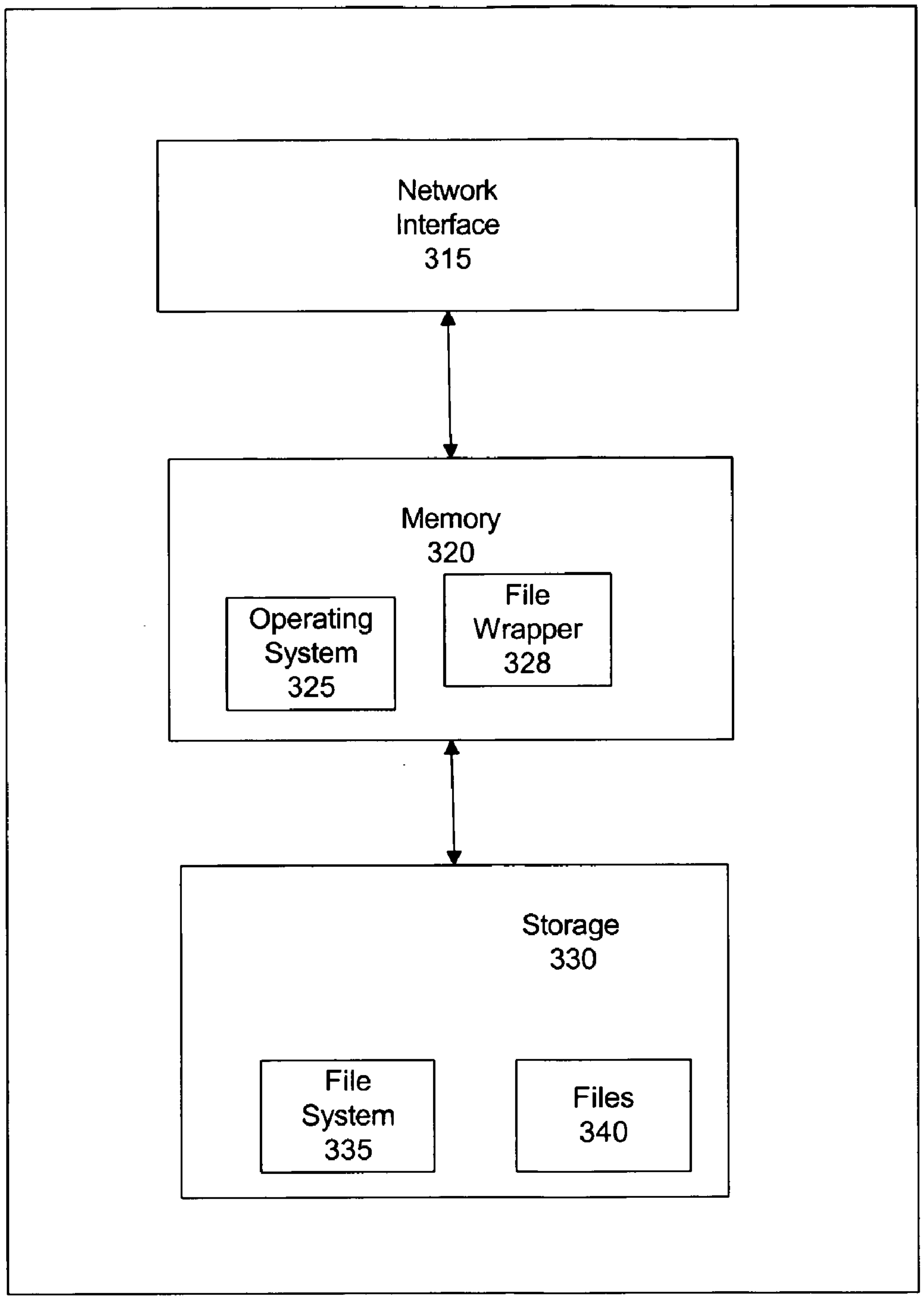

Display
240
Data Storage
230
Document
Viewer
235
Network
Interface
225
110

Fig. 2


Network Interface 315
Memory 320
Operating System 325
File Wrapper 328
Storage 330
File System 335
Files 340

140

Fig. 3

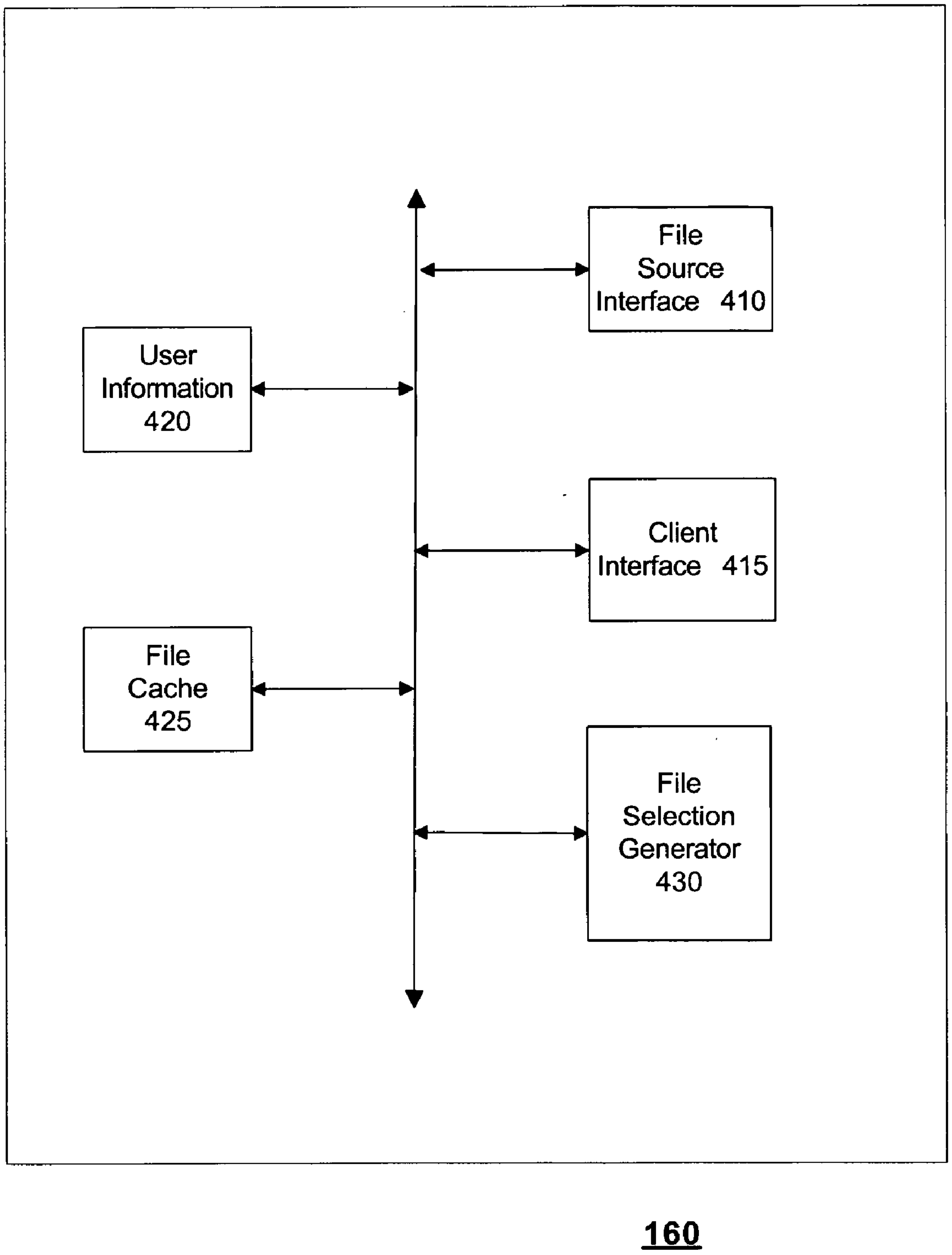

File Source Interface 410
User Information 420
Client Interface 415
File Cache 425
File Selection Generator 430
160

Fig. 4

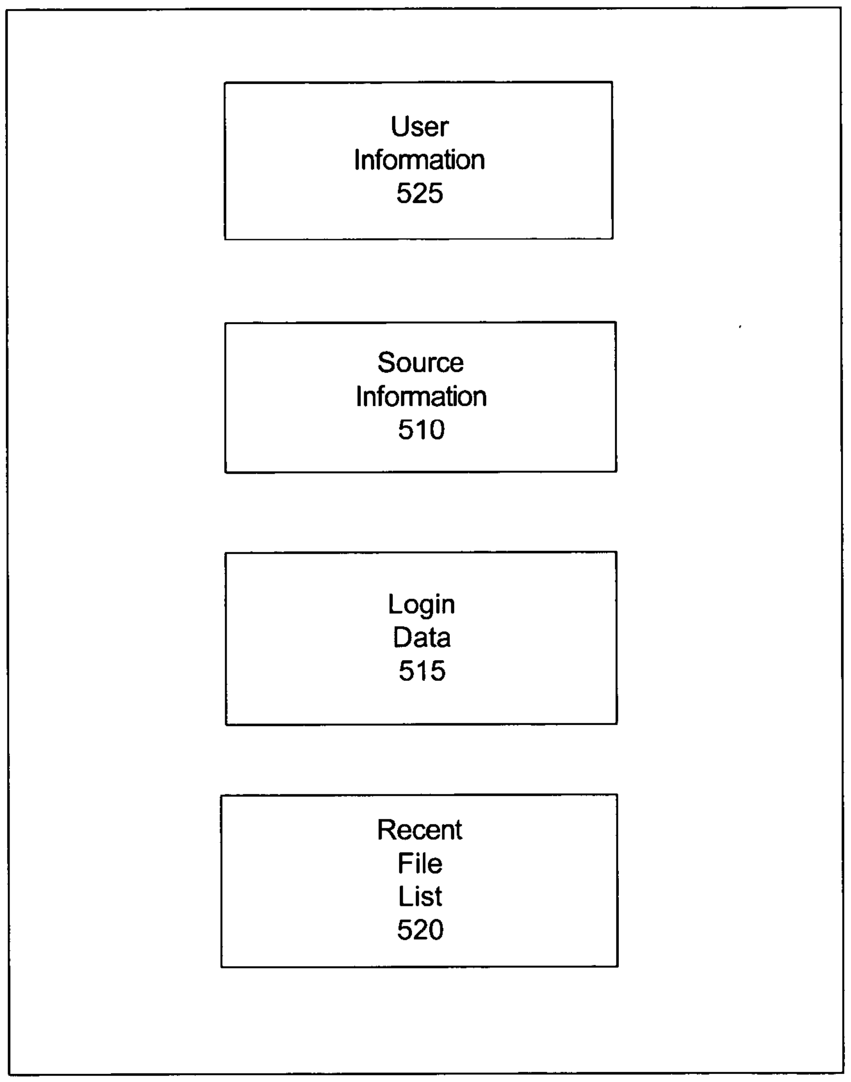

User Information 525
Source Information 510
Login Data 515
Recent File List 520
500

Fig. 5

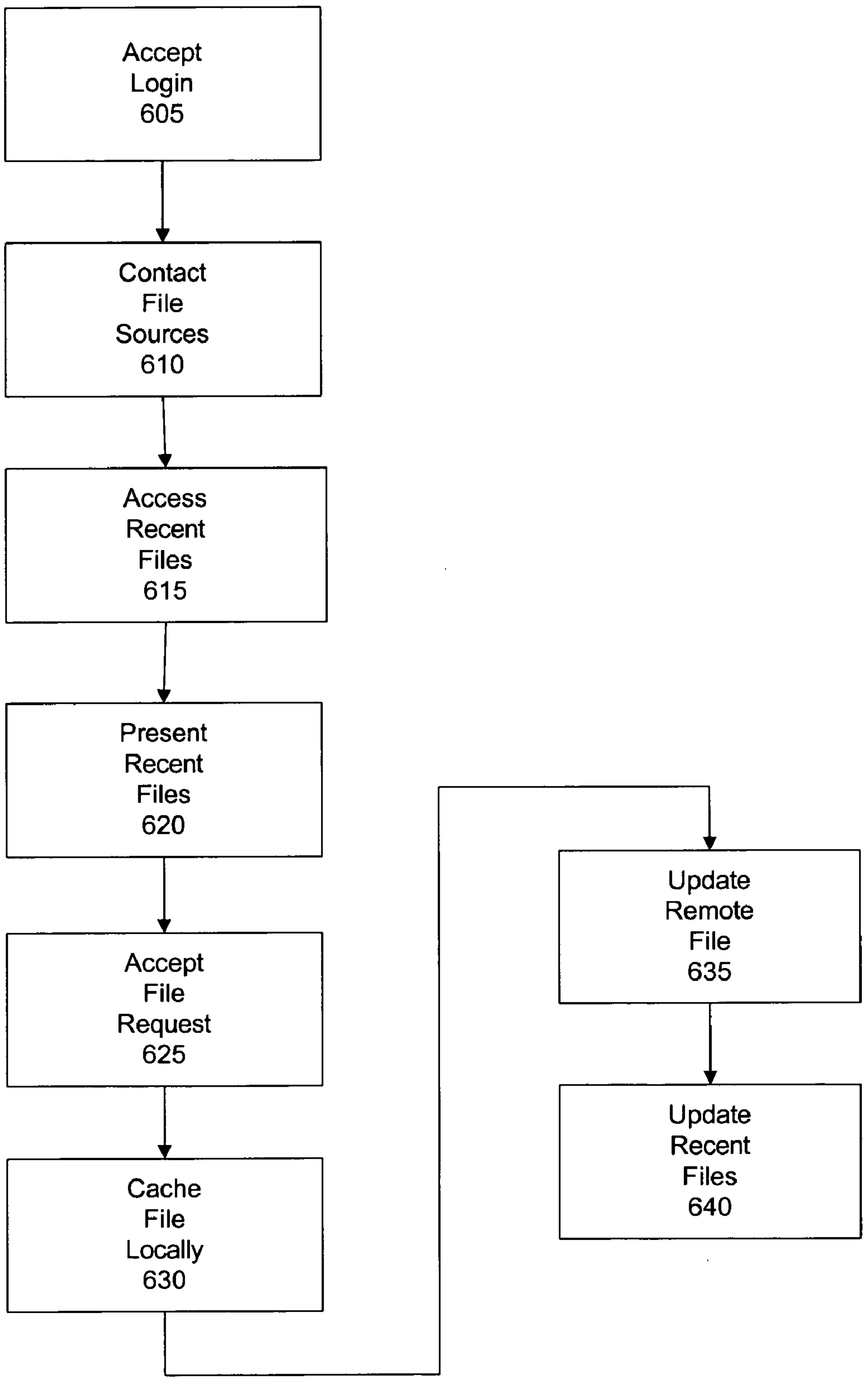

Accept Login 605
Contact File Sources 610
Access Recent Files 615
Present Recent Files 620
Accept File Request 625
Cache File Locally 630
Update Remote File 635
Update Recent Files 640

Fig. 6

SYSTEM AND METHOD FOR CUSTOMIZED DOCUMENT SELECTION

FIELD OF THE INVENTION

[0001] The current invention relates generally to remote file access and particularly to systems and methods for enabling access to multiple file sources.

BACKGROUND OF THE INVENTION

[0002] In the present business environment users are increasingly dependent on access to electronic documents and other files for performing regular business functions. Historically, users stored files upon their local machines and when traveling stored the files on portable media. However, this practice was less than ideal as it did not allow multiple users to simultaneously access the newest version of a file, the size of the physical media limited the types of files that could be used, and the physical media were often unreliable.

[0003] As networking technology became more easily accessible, some solutions arose to allow users to access their files away from their home or work computers. Several operating systems enabled users to access files stored on remote networks through internet gateways or TCP/IP enabled file sharing. However, security concerns often limited the usability of such solutions. In order to preserve the vital integrity of the files stored on local networks, such solutions often require users to employ client side Virtual Private Networking (VPN) connections to allow secure access to their remotely stored files. While VPN provides a tolerable level of protection to the remotely stored files, it presents a number of significant difficulties.

[0004] Firstly, configuring a client to use VPN requires both administrator level access to the client machine and a lengthy setup procedure. For situations where users wish to access their files from public machines such as those at a retail business center or an internet cafe, this proves to be an intolerable inconvenience. Additionally, many client devices that have entered the market during the past few years such as data-ready cellular phones and Personal Data Assistants (PDA), lack robust support for VPN. Furthermore, VPN typically only allows access to one private network at a time, requiring users that seek to access multiple secure networks to disconnect from the first network before accessing the second network, rather than enjoying simultaneous access to both networks.

[0005] Additionally, beyond their restrictions in accessing the raw data sources, VPN and similar remote-access solutions often fail to provide a useful interface for accessing the stored files. Users are forced to wade through complicated file hierarchies to access their preferred files, a task which can be especially cumbersome when using slower devices.

[0006] What is needed is a solution that allows users easy and secure access to frequently used files on a variety of file sources.

SUMMARY OF THE INVENTION

[0007] An access server allows secure access to commonly used files stored on multiple file sources from a variety of client devices. The access server stores user identification and password information for multiple file sources. The access server is connected to multiple file sources across a Local Area Network (LAN) or Wide Area Network (WAN). Additionally, the server maintains a web gateway for access by client devices.

[0008] Upon receiving a login request from a user, the access server extracts a list of file sources associated with the user, and accesses each of those sources. The server contacts each source and submits identification and password information associated with each source.

[0009] The server is then configured to extract a list of the most recently used files by the current user as well as information associated with those files. The server presents to the user an interface including links to his most recently used files.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] **FIG. 1** illustrates the interaction among clients, file sources, and an access server in accordance with one embodiment of the present invention.

[0011] **FIG. 2** is a closer view of a client system in accordance with one embodiment of the present invention.

[0012] **FIG. 3** is a closer view of a file source in accordance with one embodiment of the present invention.

[0013] **FIG. 4** is a closer view of a memory of an access server in accordance with one embodiment of the present invention.

[0014] **FIG. 5** is a closer view of a user file record in accordance with one embodiment of the present invention.

[0015] **FIG. 6** is a flow chart illustrating one embodiment of a method for retrieving and editing a file stored on a remote file source.

DETAILED DESCRIPTION

[0016] Users often store documents such as memos, spreadsheets, and presentations on networked file servers in their homes or offices.

[0017] **FIG. 1** illustrates the interaction among clients, file sources, and an access server in accordance with one embodiment of the present invention. An access server **125** is in communication with a group of file sources **140**, **145**, **150** and a client system **110**. The access server **125** is a server having memory **160**, and a network interface **165** that is configured to enable secure access to the file sources **140**, **145**, **150** from the client system **110**.

[0018] The file sources **140**, **145**, **150** are file sources such as stand-alone file servers, desktop computers, laptop computers, or any other system upon which files are stored. The file sources **140**, **145**, **150** may employ any of a number operating systems, including but not limited to: Windows 2000, Windows XP, Linux, Solaris, Netware, or Linux. The access server **125** accesses the file sources **140**, **145**, **150** via its network interface **165**, through a LAN, WAN, or customized dialup connection. The access server **125** may be located behind whatever firewall protections the file sources may use. Alternately, the access server **125** may use VPN or a customized connection mechanism to reach the file sources **140**, **145**, **150**. The connection mechanism is preferably modular and thus transparent to the client system **110**.

[0019] The client system **110** is a device remote to the access server **125** used to access, manipulate, print, and/or view the files stored on the file sources **140**, **145**, **150**. The client system **110** can be a personal computer, personal data assistant, or any device having file-viewing capacity. The client system **110** establishes a network connection between itself and the access server **125** and views the files on the file sources **140**, **145**, **150** through an interface generated by the client or the access server **125**. In one embodiment this interface is a web interface. In an alternate embodiment, the access server **125** generates a customized interface according to the viewing capacities of the client system **110**, or the client system generates the interface itself.

[0020] **FIG. 2** is a closer view of a client system **110** in accordance with one embodiment of the present invention. The client system **110** includes data storage **230**, a document viewer **235**, and a network interface **225**. The data storage **230** stores documents and other necessary data locally on the client system **110**. The data storage can be a solid state device such as a hard drive. Alternately the data storage can be Static Random Access Memory (SRAM) or Dynamic Random Access Memory (DRAM).

[0021] The network interface **225** manages communication between the client device **110** and remote systems. The network interface **225** includes hardware communication devices such as a modem, Ethernet, or WiFi communicator as well as software protocols for managing communication.

[0022] The document viewer **235** parses documents received via the network interface **225** and presents them upon the display **240**. The document viewer **235** preferably includes the ability to interpret HTML as well as most conventional document formats (Microsoft Word, Microsoft Excel, etc.) In one embodiment, the document viewer **235** is an application selector routine, that upon receiving a file across the network interface **225**, determines a preferred application for viewing the file type and submits the file to the appropriate application, which then displays the document on the display **240**. The document viewer can be a standardized application such as a web browser or an application specially configured for use with the access server **125**. In an alternate embodiment, the document viewer **235** is a single application capable of displaying multiple file formats which independently receives and displays any received documents. In some embodiments, the document viewer **235** includes functionality for editing any received documents and either storing them in the data storage **240** or transmitting them back to the access server **125**.

[0023] In some embodiments, the client system **110** is not configured to actually view files maintained by the access server **125**. For example, if the client system **110** were a cell phone without significant display functionality, the client system could copy the files to remote locations, direct the access server to print them at remote printers, and perform other functions, but not actually view the files themselves.

[0024] In some embodiments, the access server **125** or the client itself provides filters to recently used file lists according to the capacities or primary uses of the client **110**. For example, if the client **110** were used primarily to generate video presentations, the file source wrapper would generate a list of the most recently used video presentation. Alternately, if the client couldn't view Microsoft Word documents or files larger than a predetermined size, the access server **125** or client itself **110** could filter larger files or Microsoft Word documents.

[0025] **FIG. 3** is a closer view of a file source **140** in accordance with one embodiment of the present invention. The file source preferably includes a network interface **315**, a memory **320**, and a storage **330**. The file source can be a personal computer, a file server, or a public use computer.

[0026] The network interface **315** maintains communication between the file source **140** and any devices attempting to access the storage **330**. The network interface can be an Ethernet connection, modem, WiFi transmitter, or any hardware capable of communicating with outside devices.

[0027] The memory **320** stores data in temporary use and maintains the operating system **325** which regulates access between the storage **330** and the network interface **315**. The operating system **325** can include user identifiers and passwords which are used to regulate access to the storage **330**. In one embodiment, the operating system **325** maintains an account for at least one user and restricts access to certain sections of the storage **330** to that user.

[0028] The storage **330** includes a file system **335** which maintains organizational information for the files **340** stored on the storage **330**. The file system **335** maintains a directory structure, a time of last use for each of the files **340**, access permissions for each of the directories **340**, and all other information needed to properly manage access to the files **340** by the operating system **325**.

[0029] The storage also includes a list of recently accessed files **345**, which is maintained by the operating system **345**. In one embodiment, the recent files **345** are a list of the files most recently accessed, modified, or created, by the operating system **325**. In an alternate embodiment, the operating system **325** maintains a separate recent file list **345** for each application which is run upon the system. For example, the operating system could maintain a list of the files most recently used by Microsoft Word, a separate list of the files last used by Adobe Acrobat, and a third list of the files last used by the Windows Media Player. The number and details of the lists **345** are governed by the complexity of the operating system **325**.

[0030] In one embodiment, the file source **140** includes a file source wrapper **328**. The file source wrapper **328** is a module which is configured to manage interaction with the access server **125**. The file source wrapper **328** periodically gathers lists of recently used files to present to the access server **125** when it contacts the file source. The file source wrapper may draw the list from the operating system **325** or check lists maintained by different applications on the file source **325** and unify them to a single file list. In some embodiments, a single file wrapper **328** acts as a gateway for a number of file sources behind a firewall.

[0031] In additional embodiments, the file source does not include a file source wrapper **328** and the access server **125** logs in and interacts with the file source through a traditional client server process maintained by the operating system **325**.

[0032] **FIG. 4** is a closer view of the memory **160** of an access server **125** in accordance with one embodiment of the present invention. The memory **160** includes a number of modules, specifically, a file source interface **410**, a client interface **415**, user information **420**, a file cache, **425**, and a file selection generator **430**, each of which provides some functionality for the access server **125**. The modules **410**, **415**, **420**, **425** can be hardware, software, firmware, or any combination thereof.

[**0033**] The user information **420** stores customized user information for each of a number of users of the access server. The user information **420** stores a list of sources for each user, a list of usernames and passwords for each source, and a list of recent files associated with each source.

[**0034**] The file source interface **410** manages interaction between the access server **125** and the file sources **140**, **145**, and **150**. The file source interface **410** can include a standardized API that interfaces with the file source wrappers **328** on the file sources through a single standardized API provided by each of the file source wrappers, or customized front ends that are configured to interface with the file sources. The file source interface **410** is configured to receive general access instructions from the other modules and translate them to the format of the file source **140**, **145**, **150**. For example, upon receiving a request for a file owned by user A and located on a Linux server, the file source interface would log into the Linux server, submit user A's ID and password information, log into the Linux server, navigate to the correct directory, and retrieve the file to the access server **125**. Alternately, it could send the request to the file source wrapper located on the file source, which would itself locate the file and transmit it.

[**0035**] The client interface **415** generates a customized interface for the client system **110**. This interface preferably includes a listing of recently used files, and the ability to view and edit the files, either through capacities internal to the interface or by utilizing file viewers on the client system **110** itself. The client interface **415** receives commands from the client, translates them and passes them to the appropriate module. In one embodiment, the interface generated by the client interface **415** is a standard HTML interface. In an alternate embodiment, the client interface, upon initially being contacted by the client system, **110**, determines its identity and capacities, and selects the interface best suited for the client system. For example, if the client system **110** were a cell phone, the client interface would generate a low bandwidth interface. In some embodiments, the access server **125** provides a standardized API for interacting with the file list and files, and the client **110** is responsible for generating an interface based on information returned from the access server **125**.

[**0036**] The file cache **425** stores locally available versions of files that are accessed by the access server **425**. When a file is first accessed, the file source interface **410** retrieves the file from the file source **140**. Any changes are stored in the local file cache **425**. When a user attempts to view a file, the file is transferred from the file source **140** to the file cache. In one embodiment, when a session closes, the access server **125** evaluates the cache **425** to determine if any files have been changed. Any changed files are transferred back to the file source **140**. In an alternate embodiment, changed files are continuously updated on the file source **140**. For example, if one of the file sources was a laptop or another intermittently connected source, the access server would check all cached files to determine whether any had changed since the last connection, and update them on the file source **140**.

[**0037**] The file selection generator **430** contacts each of the file sources **140**, **145**, **150** in order to generate a list of recently used files for each user. The recently used files may comprise lists of recently accessed, recently modified, or recently created files. In one embodiment, the file selection generator **430**, through the file source interface **410**, contacts each of the sources to receive lists of the most recently used files **345**. The file selection generator **430** can draw a single list of the most recently created, modified, or accessed files. Alternatively, the file selection generator **410** can extract a separate list of the most recently used files for each of a number of commonly used applications. For file sources that have file source wrappers, the file source wrapper generates the list and provides the list to the access server.

[**0038**] Some file sources **140** do not maintain separate lists of commonly used files. For these file sources, the file selection generator **430** checks the file system **335** of the file source **140** to determine which files were last accessed. Alternatively, the file selection generator can query the file source to determine which files were last modified or created. In some embodiments, the work of gathering and generating lists of recently used files is performed by the file source wrapper **328** which resides on the file source.

[**0039**] The number and type of files that are selected and received as recently used files can be universal, or configured separately by each user.

[**0040**] **FIG. 5** is a closer view of a user file record **500** stored in accordance with one embodiment of the present invention. The file record **500** is usually stored in the user information **420**. Typically, the user information **420** includes multiple records, each record associated with a user.

[**0041**] The user record **500** includes user identification **525**. The user identification **525** includes information used to identify the user. Upon receiving a successful login from a user, the access server **125** searches the user identification **525** sections of each user record **500** until it finds a user ID corresponding with the submitted login information.

[**0042**] The source information **510** stores an identifier for each of the file sources **140**, **145**, **150**. The identifier can include the Internet Protocol (IP) address, hostname, or any other usable identifier. This information is used to make contact with the file sources **140**, **145**, **150**. Additionally, the source information **510** stores the operating system and computer type for the file sources **140**, **145**, **150**. This information is utilized by the file source interface **410** to determine how to access and navigate the file sources **140**, **145**, **150**.

[**0043**] The login data **515** stores usernames and passwords for each of the file sources **140**, **145**, **150** listed in the source information **510**. The file source interface uses this information to access the file sources **140**, **145**, **150**. In some emdodiments, the access server **125** does not maintain usernames and passwords for all file sources and instead prompts the user for credentials when accessing the file source **140**.

[**0044**] The recent file list **520** is a list of the files most recently accessed by the user. The recent file list is periodically updated by the file selection generator **430**. In one embodiment, the recent file list **520** is updated whenever the user logs into the access server **125**.

[**0045**] **FIG. 6** is a flow chart illustrating one embodiment of a method for retrieving and editing a file stored on a remote file source **140**. It should be noted that the functions performed in this figure can be shifted between the client **110**, access sever **125**, or file sources **140, 145, 150**. For example, the file sources may include a file source wrapper which gathers information about files stored on the file source and passes it to the access server. Alternately, the clients may perform filtering operations on recent file lists passed from the access server. The process begins with the access server accepting **605** a user login via the client interface **415**. Upon receiving the user login, the access server **125** searches the user identification **525** sections of each user record **500** until it finds a user ID corresponding with the submitted login information.

[**0046**] The access server **125** then reads the source information **510** and contacts each of the sources **140, 145, 150**. If the user record **500** includes the login and password information for each of the sources, the access server extracts it. For any sources for which the login information is not present, the access server **125** prompts the user to provide username and password information. The file source interface **410** uses the source information **510** to manage its interaction with the file sources **140, 145, 150**. The file selection generator **430** then accesses **615** the recent files from each of the sources. In one embodiment, the file selection generator **430**, through the file source interface **410** contacts each of the sources to receive lists of the most recently used files **345**. The file selection generator **430** can draw a single list of the most recently accessed files. Alternatively, the file selection generator **410** can extract a separate list of the most recently used files for each of a number of commonly used applications. The file selection generator **430** is configured to check the client interface for a list of applications accessible by the user and present recently used files for each of those applications. For operating systems that lack recent file histories, the file selection generator **430** sorts all of the files accessible to the user and accepts those having the latest access date as the most recent files. For those file sources **140** that maintain file source wrappers **328**, the file selection generator contacts the file source wrapper **328** and obtains the recent file information. The file source wrapper, access server, or client may filter the files according to the viewing capacities of the clients or other criteria. This filtering can include filtering by file type, file size, by time accessed, or any other relevant criteria.

[**0047**] The client interface **415** then presents **620** the user with links to the most recently accessed files. In one embodiment, the client interface provides a list of recently accessed files as part of a startup screen without additional prompting or requests. As used herein the term "link" refers to a graphical icon or text grouping that when selected via the user interface causes a document or folder to be accessed. In one embodiment, the client interface **415** also provides navigational links for accessing folders and drives on the file sources **140, 145, 150**.

[**0048**] A user then selects a file through the client interface **415**, which receives **625** the file request and retrieves the file from the file source **140**. The access server **125** then caches the file locally in the file cache **425** so that it can be edited. The client system **110** may also cache a copy of the file locally. Once the file has been updated and the user submits a command to save the file, the cached file is copied **635** to the file source **140**, which then updates the file in the remote directory.

[**0049**] The file selection generator **430** then updates **640** the recent file list **520** of the user to indicate that the accessed file is a recently used file. If the file was originally selected from the recent file list **520**, the file selection generator **430** places the file at a higher level of recency. If the file was not originally on the recent file list **520**, it is added to the list as the most recently used file.

[**0050**] Other features, aspects and objects of the invention can be obtained from a review of the figures and the claims. It is to be understood that other embodiments of the invention can be developed and fall within the spirit and scope of the invention and claims.

[**0051**] The foregoing description of preferred embodiments of the present invention has been provided for the purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise forms disclosed. Obviously, many modifications and variations will be apparent to the practitioner skilled in the art. The embodiments were chosen and described in order to best explain the principles of the invention and its practical application, thereby enabling others skilled in the art to understand the invention for various embodiments and with various modifications that are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the following claims and their equivalence.

[**0052**] In addition to an embodiment consisting of specifically designed integrated circuits or other electronics, the present invention may be conveniently implemented using a conventional general purpose or a specialized digital computer or microprocessor programmed according to the teachings of the present disclosure, as will be apparent to those skilled in the computer art.

[**0053**] Appropriate software coding can readily be prepared by skilled programmers based on the teachings of the present disclosure, as will be apparent to those skilled in the software art. The invention may also be implemented by the preparation of application specific integrated circuits or by interconnecting an appropriate network of conventional component circuits, as will be readily apparent to those skilled in the art.

[**0054**] The present invention includes a computer program product which is a storage medium (media) having instructions stored thereon/in which can be used to program a computer to perform any of the processes of the present invention. The storage medium can include, but is not limited to, any type of disk including floppy disks, optical discs, DVD, CD-ROMs, microdrive, and magneto-optical disks, ROMs, RAMs, EPROMs, EEPROMs, DRAMs, VRAMs, flash memory devices, magnetic or optical cards, nanosystems (including molecular memory ICs), or any type of media or device suitable for storing instructions and/or data.

[**0055**] Stored on any one of the computer readable medium (media), the present invention includes software for controlling both the hardware of the general purpose/specialized computer or microprocessor, and for enabling the computer or microprocessor to interact with a human user or other mechanism utilizing the results of the present invention. Such software may include, but is not limited to, device drivers, operating systems, and user applications.

[0056] Included in the programming (software) of the general/specialized computer or microprocessor are software modules for implementing the teachings of the present invention.

What is claimed:

1. A method for providing access to remotely stored files, the method comprising:

receiving an identifier from a user;

contacting at least one file source associated with the identifier; and

generating a list of at least one recently accessed file from the file source.

2. The method of claim 1, further comprising presenting the user with an option of printing the file.

3. The method of claim 1, further comprising retrieving the file from the file source.

4. The method of claim 1, wherein generating the list of at least one recently accessed file comprises receiving a list of at least one recently accessed file on the file source.

5. The method of claim 1, wherein generating the list of at least one recently accessed file comprises:

reading a time of last access for files stored on the file source; and

selecting a file according to its time of last access.

6. The method of claim 1, wherein generating the list of at least one recently accessed file comprises:

determining an application available to the user; and

receiving from the file source a list of at least one file associated with the application.

7. The method of claim 1, further comprising:

receiving changes to a copy of the file; and

transmitting the copy of the file to the file source.

8. A system for providing access to remotely stored files, the system comprising:

a client interface configured to receive an identifier from a user; and

a file selection generator configured to:

contact at least one file source associated with the identifier; and

generate a list of at least one recently accessed file from the file source.

9. The system of claim 8, wherein the client interface is further configured to present the user with an option to view the file.

10. The system of claim 9, wherein the list of recently accessed files comprises a list of most recently created files.

11. The system of claim 9, wherein generating the list of at least one recently accessed file comprises receiving a list of at least one recently accessed file on the file source.

12. The system of claim 9, wherein generating the list of at least one recently accessed file comprises:

reading a time of last access for files stored on the file source; and

selecting a file according to its time of last access.

13. The system of claim 8, wherein generating the list of at least one recently accessed file comprises:

determining an application available to the user; and

receiving from the file source a list of at least one file associated with the application.

14. The system of claim 8, wherein the client interface is configured to receive changes to a copy of the file.

15. A computer program product, stored on a computer readable medium, and including computer executable instructions for controlling a processor to provide access to remotely stored files, the instructions comprising:

receiving an identifier from a user;

contacting at least one file source associated with the identifier; and

generating a list of at least one recently accessed file from the file source.

16. The computer program product of claim 15, wherein the computer code instructions further comprise providing the user with an option of transferring the file to a system of the user or to another location.

17. The computer program product of claim 15, wherein the list of at least one recently accessed file comprises a list of most recently modified files.

18. The computer program product of claim 15, wherein generating the list of at least one recently accessed file comprises receiving a list of at least one recently accessed file on the file source.

19. The computer program product of claim 15, wherein generating the list of at least one recently accessed file comprises:

reading a time of last access for files stored on the file source; and

selecting a file according to its time of last access.

20. The computer program product of claim 15, wherein generating the list of at least one recently accessed file comprises:

determining an application available to the user; and

receiving from the file source a list of at least one file associated with the application.

21. The computer program product of claim 15, wherein the instructions further comprise:

receiving changes to a copy of the file; and

transmitting the copy of the file to the file source.

22. A system for providing access to files, the system comprising:

a file server storing files and a list of at least one recently accessed file; and

an access server configured to:

receive an identifier from a user;

contact the file server in response to receiving the identifier; and

retrieve the list of at least one recently accessed file from the file server.

23. The system of claim 22, wherein the access server is further configured to retrieve the recently accessed file from the file server.

24. The system of claim 22, wherein the access server is further configured to provide the user with an option of emailing the file to another location.

25. The system of claim 22, wherein the file server stores a list of at least one recently used file associated with an application.

26. A method for enabling a user to access and manage files from a plurality of file sources, the method comprising:

receiving an identifier from the user;

contacting at least two file sources; and

generating a unified list of recently accessed files from the at least two file sources.

27. The method of claim 26, further comprising presenting the user with an option of faxing a file in the unified list.

28. The method of claim 27, further comprising retrieving the file from the file source.

29. The method of claim 25, wherein generating the unified list comprises receiving a list of recently used files from a file source.

30. The method of claim 25, wherein generating the unified list comprises:

determining an application available to the user; and

receiving from a file source a list of at least one file associated with the application.

31. The method of claim 25, further comprising presenting to the user a startup screen which includes a visual display representing the list of recently accessed files.

32. A computer program product, stored on a computer readable medium, and including computer executable instructions for controlling a processor to provide access to enable a user to access and manage files from a plurality of file sources, the instructions comprising:

receiving an identifier from a user;

contacting at least two file sources; and

generating a unified list of recently accessed files from the at least two file sources.

33. The computer program product of claim 32, wherein the computer code instructions further comprise providing the user with an option of viewing a file.

34. The computer program product of claim 33, wherein the computer code instructions further comprise retrieving the file from the file source.

35. The computer program product of claim 32, wherein generating the list of at least one recently accessed file comprises:

determining an application available to the user; and

receiving from the file source a list of at least one file associated with the application.

36. The computer program product of claim 32, wherein the instructions urther comprise:

receiving changes to a copy of the file; and

transmitting the copy of the file to the file source.

37. The computer program product of claim 32, wherein the instructions further comprise:

receiving changes to a copy of the file; and

transmitting the copy of the file to the file source.

38. The computer program product of claim 32, wherein the instructions further comprise presenting to the user a startup screen displaying the unified list.

* * * * *